FlowMenu: Combining Command, Text, and Data Entry

François Guimbretière, *Terry Winograd*
Computer Science Department
Stanford, CA 94305-9035
+1 (650) 723 0618
{francois,winograd}@cs.stanford.edu

ABSTRACT
We present a new kind of marking menu that was developed for use with a pen device on display surfaces such as large, high resolution, wall-mounted displays. It integrates capabilities of previously separate mechanisms such as marking menus and Quikwriting, and facilitates the entry of multiple commands. While using this menu, the pen never has to leave the active surface so that consecutive menu selections, data entry (text and parameters) and direct manipulation tasks can be integrated fluidly.

KEYWORDS: Quikwriting, Marking menu, Control Menu, Interactive surface

INTRODUCTION
The *FlowMenu* is a command-entry system well suited for interactive display surfaces with pen-based input. It was designed for the Interactive Mural [5], a custom-built, large, high resolution, wall-mounted display, but FlowMenu can be used with any device that accepts stylus input (direct or indirect). Several stylus command-entry approaches have been proposed previously, but none of them smoothly integrates command selection, text entry and direct manipulation. Separating the command selection and parameter entry mechanisms in a pen-based display can be a major drawback for power users, who exhibit a pattern of rapid shifting between keyboard and mouse. The FlowMenu extends Perlin's Quikwriting technique [10] to a full hierarchical menu system to alleviate this shortcoming. FlowMenu provides menu selection, text entry, and parameter adjustment in an integrated mechanism, delivering a smooth and efficient interaction for experienced users while providing a learning path for novice users.

PREVIOUS WORK
Many menu systems offer some form of context (or pop-up) menus, which enable users to make a menu selection at the point of focus rather than in a distant menu bar. These are especially effective while interacting on large surfaces.

Pie menus [1] use radial rather than linear selection, which enables the user to learn directional gestures so selection can be made without needing to look at the menu items. For a more fluid interaction, the menu is not even drawn if a gesture is made immediately upon menu activation. Pie menus can be hierarchical, with the end of one stroke becoming the beginning point for a secondary selection. Users can learn zigzag gesture patterns as single elements, which correspond to a multi-level selection.

Marking menus [8, 6] are an extension of pie menus, in which the shape of the path (which is a visible trail on the screen) is interpreted as a selection if appropriate, or can be recognized as another gesture. They can be more flexible since recognition is done on the shape rather than the precise positioning of the turning points of the path [1]. Learning is facilitated by the fact that if the user pauses while marking, the underlying menu hierarchy is displayed. Marking menus provide a way to merge object selection, tool selection and direct manipulation. Because marks are normally terminated by a pen-up transition, the discrimination between mark and direct manipulation is sometimes difficult to identify, requiring a timeout [7]. The earlier Momenta [12] system provided a similar capability with more limited scope, called the "command compass".

Pook [11] introduced control menus, an extension of marking menus, to allow the entry of continuous parameters, such as distances and zoom control, associated with a command selection. The system relies on an arbitrary threshold distance to distinguish between marking and direct manipulation.

Many designs have been proposed for entering text with a pen device using a circular layout: T-cube [13], Quikwriting [10] and Cirrin [9]. We will only describe the last two of these here. Quikwriting [10] was developed as a way to improve text entry on small hand held devices such as the Palm Pilot. It can be seen as a specialized form of marking menu, in which the end of a selection gesture is not indicated by lifting the button or pen (as it is for marking menus), but by returning to a home position. The user never has to remove the pen from the surface in entering a series of characters, so common sequences become learned as unitary complex gestures. Quikwriting dealt strictly with text entry, and one aspect of FlowMenu is extending the sequential input of mul-

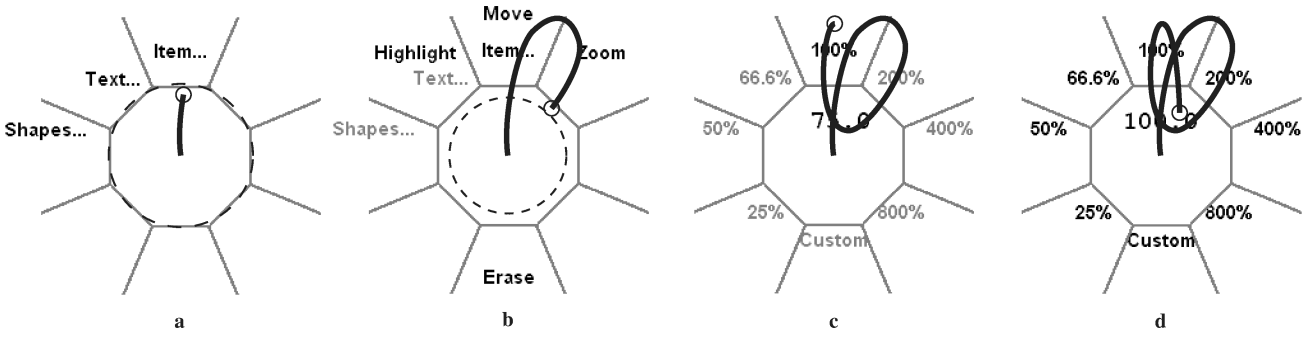

Figure 1: To zoom, the user moves the pen from the rest area into the *Items...* octant (**a**). Submenus (*Highlight*, *Move*, *Zoom*) appear and the first level menu items not selected are grayed out (**b**). Entering the *Zoom* octant submenu, then moving back to the rest area dismisses the root level menu and brings up the zoom menu with the current zoom value (75%) displayed in the center (**c**). A new zoom value of 100% is selected by moving into the octant for the desired value and back to the center at which point the zoom is applied (**d**). Several zoom values can be tried out during the same interaction since the zoom menu stays in place until the pen is lifted. The dashed circles added to the illustration (**a**) and (**b**) show the transition boundaries for leaving and entering the rest area (see text). For explanatory purposes, the figures in this paper explicitly show the pen track and the underlying selected object is shown only in Figure 3. In normal use, the pen track is not displayed and the selected object is visible behind the transparent menu.

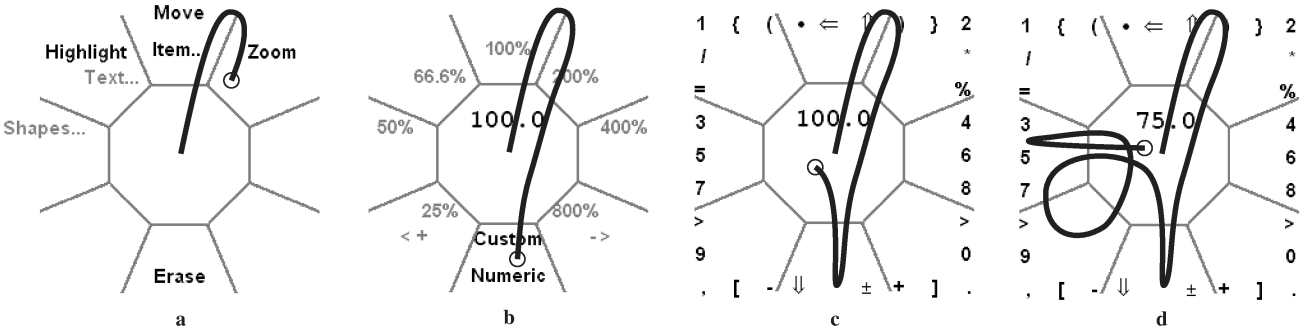

Figure 2: After selecting *Item...* → *Zoom* from the root menu (**a**), the user selects *Numeric* to enter the new zoom value as a sequence of digits (**b**). The zoom menu is dismissed and the Quikwriting system is brought up (**c**) so that she can enter the zoom value (**d**).

tiple items to a menu system as well. Cirrin [9] is a soft keyboard in which letters are arranged at the circumference of a circle. Like Quikwriting it provides a way to enter successive letters of a word in a continuous stroke without having to lift the pen. After an initial training period, words can be remembered as a kind of shorthand. The initial layout of 26 primary entries without hierarchy makes it less convenient to extend to a menu system.

THE FLOWMENU

The FlowMenu is presented as a radial menu with 8 octants and a central rest area (figure 1). Starting from the rest area, the user selects a top-level menu item by entering the corresponding octant. As she does, sub-menus for this menu appear laid out further away from the center while non-selected top-level items are grayed out. Moving the pen to the submenu octant and reentering the rest area from this octant will trigger menu selection. The user can abort the interaction by removing the pen from the surface before reentering the rest area. With a simple FlowMenu, the user can access 8 top-level menu items, each with 8 submenu items. However since each selection of a menu ends with the cursor at the center of the menu, successive menu interactions can be merged together to build deeper hierarchies and arbitrarily long sequences of interactions. Figure 1 show an example where after selecting the zoom submenu from the system menu, the system menu is removed and the zoom menu is brought up to let the user adjust the zoom.

Merging menu selection and parameter entry is easy because commands are segmented by the return of the cursor to the rest area. To let the user enter an alphanumerical value after a menu selection we remove the menu from the screen and present in its place a Quikwriting pad. Figure 2 shows such an interaction. The selection *Item...* → *Zoom* → *Numeric* brings up the Quikwriting system to let the user enter a numeric zoom value. The user can learn a composite sequence of commands and text as the superposition of simple loop gestures such as shown in figure 2d. The system can also be used in a way similar to control menus by letting the

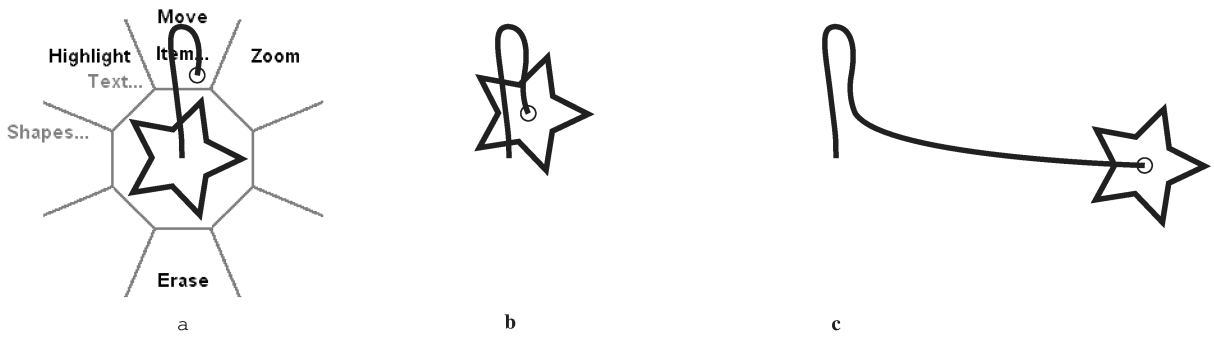

Figure 3: FlowMenu interaction integrates smoothly with direct manipulation. Here after selecting the move action from the root menu (**a**), the user continues directly with the drag interaction (**b,c**). In contrast to marking menus, the selected object follows the cursor immediately. The initial jump of the object from the center of the menu to the beginning of the drag interaction (**b**) has not been a problem in practice since during a drag, users focus their attention on the target location [2].

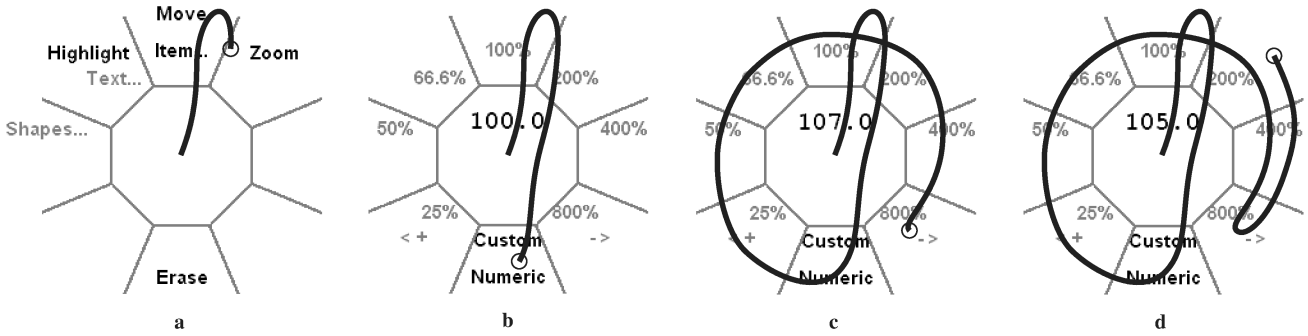

Figure 4: In this example, the knob interaction is used to adjust the zoom level. After selecting *Item... → Zoom → Numeric* (**a,b**), the user circles the pen around the center area, using the menu as a knob for fine adjustment. Each time an octant line is crossed, the value is incremented by a small amount (**c**) (decremented if counter-clockwise (**d**)). The zooming is done real-time, with the object visible (omitted in this figure for clarity) so visual feedback is provided at all times.

user perform a drag after the action selection, as shown in figure 3. The return to the rest area obviates the need for an arbitrary threshold distance or time-out to distinguish marking and direct manipulation.

Finally, the FlowMenu can be used in a "knob" mode in which the user interacts with the menu as though it was a knob. As shown in figure 4 crossing a octant line clockwise (resp. counter-clockwise) increases (resp. decreases) the value by a small amount. This kind of interaction is very useful for dynamically fine-tuning parameter values such as zoom level.

CURRENT USE

FlowMenu is the default menu system for the Interactive Mural. The Mural uses the ultrasonic EFI EBeam [3] system to track the pen on the screen surface. The root menu mechanism is triggered by depressing the system menu button while the pen is touching the surface. The menu button can be released as soon as the menu interaction is initiated. Removing the pen from the surface before reentering the rest area will abort the interaction. Pressing the menu button at any time will abort the current interaction and bring the user back to the root of the menu hierarchy. Since the current EBeam pens do not provide a menu activation button, we use the button of a separate device (a wireless mouse held in the user's other hand) as a proxy. We plan to integrate the menu button into the physical pen device in the next version of our system, before extended user testing.

All of the interaction modes described above have been used in a tool for entering and manipulating simple hand-drawn sketches during a brainstorming session. In this application, FlowMenu allows users to move and zoom sketches and to enter labels. We have only informal experience to date, and will do user testing later this year during the evaluation of our brainstorming tool.

Note that unlike marking menus [6], we did not implement a delay in the appearance of the menu. Given the characteristics of our toolkit (use of transparency, high speed rendering and decoupled rendering and interaction loops) there seems to be no disadvantage to displaying them immediately even if the user is making a coordinated combination gesture. While immediate menu appearance has the potential for visual distraction, we conjecture that user testing will show that expert users are not distracted, and that novice users will benefit from the absence of a time-out pause.

DISCUSSION

The FlowMenu shares the advantages of both marking menus and Quikwriting. Object and verb selection are combined into a single operation, and sequences of verb selection can be chunked as a complex ideogram-like mark. FlowMenus offer a path for learning strokes by providing an underlying self-revealing menu hierarchy to help the user in the transition from recognition to recall. Quikwriting's "return to the central rest area" style of command segmentation provides a smoother way to distinguish between menu selection and direct manipulation interaction. This segmentation makes it possible to integrate alphanumerical text entry as part of a menu interaction.

Currently, our implementation uses a simple Distance/Angle transition detection mechanism. It requires a well-calibrated input stream because the user has to leave and enter the rest area using somewhat narrow corridors. Figure 1 shows that the circle marking the return to the rest area is slightly smaller that the circle marking the departure from the rest area. We added this hysteresis to alleviate noise problems inherent to tracking devices on large surfaces. "Eyes free" interactions are limited to simple sequences like *move* (figure 3) or *zoom 100%* (figure 1). A stroke-feature based implementation will provide more robust recognition, supporting "eyes free" operation for more complex interactions.

Visual obstruction by the hand is an issue shared by all radial menus, including FlowMenus. It is a pressing problem for beginners who sometimes need to move their hand to see menu items, which can be awkward to accomplish while keeping the pen tip on the board. Our solution to this problem was to avoid using the Southeast octant (or Southwest octant for left handed use) or to use it as the opposite slot for complementary items [4]. Note that this problem only occurs in direct pen interaction systems, not in the case of indirect interaction (e.g., tablets separated from the display).

Activation of the menu near the border is a problem for all kinds of pop-up menus. Since our system uses direct interaction on the display surface, it is impossible to use any kind of cursor warping. Given the size of our screen and our particular tracking technology, we can provide a tracking area larger than the display area, allowing the user to complete a gesture even when only part of the menu is visible. This solution will not work for small devices, in which case Kurtenbach's "pull out mark" [6, section 6.2.3] can be used. Note that in our case, since successive levels of the hierarchy will appear at the same place on the screen, the problem is simpler to manage than in the case of marking menus.

This note reflects only early results of this technique, since we have not yet collected detailed user performance data such as speed of execution and learning. We predict that performance will be similar to the marking menu, but with improved performance when mixed verb and parameter entries are needed.

ACKNOWLEDGEMENT

Ian Buck and Greg Humphreys provided the rendering library. Brad Johanson wrote the input system infrastructure. Phil Weaver from EFI provided us with a custom version of the EBeam systems. We would like to thanks Tamara Munzner for her comments and our reviewers for pointing out several pertinent references. This work was supported by grants from IBM and Philips.

REFERENCES

1. J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An Empirical Comparison of Pie vs. Linear Menus. In *Proceedings of CHI'88*, pages 95–100, 1988.

2. L. Carlton. Processing Visual Feedback Information for Movement Control. *J. of Exp. Psychology: Human Perception and Performance*, 7(5):1019–1030, 1981.

3. Electronics for Imaging, Inc. *http://www.e-beam.com/.*

4. D. Hopkins. The Design and Implementation of Pie Menus. *Dr. Dobb's Journal*, 16(12):16–26, 1991.

5. G. Humphreys and P. Hanrahan. A Distributed Graphics System for Large Tiled Displays. In *Proceedings of Visualization'99*, pages 215–227, 1999.

6. G. Kurtenbach. *The Design and Evaluation of Marking Menus*. PhD thesis, University of Toronto, 1993.

7. G. Kurtenbach and W. Burton. Issues in Combining Marking and Direct Manipulation Techniques. In *Proceedings of UIST'91*, pages 137–144, 1991.

8. G. Kurtenbach and W. Buxton. User Learning and Performance with Marking Menus. In *Proceedings of CHI'94*, pages 258–264, 1994.

9. J. Mankoff and G. Abowd. Cirrin: A World-Level Unistroke Keyboard for Pen Input. In *Proceedings of UIST'98*, pages 213–214, 1998.

10. K. Perlin. Quikwriting: Continuous Stylus-Based Text Entry. In *Proceedings of UIST'98*, pages 215–216, 1998.

11. S. Pook, E. Lecolinet, G. Vaysseix, and E. Barillot. Control Menu: Execution and Control in a Single Interactor. In *Extended Abstracts of CHI2000*, pages 263–264, 2000.

12. A. Reinhardt. First Impression: Momenta Points to the Future. *Byte Magazine*, 1991.

13. D. Venolia and F. Neiberg. T-Cube: A Fast, Self-Disclosing Pen-Based Alphabet. In *Proceedings of ACM Conference on Human Factors in Computer Systems*, pages 265–270, 1994.